CompuPro®

A *GODBOUT* COMPANY

mapFORTH

Technical Manual

STAND ALONE

FORTH OPERATING SYSTEM

mapFORTH TECHNICAL MANUAL

Hayward, CA 94545

Document #15051
Filenames: FORTH.MAN
FORTH1.MAN

First Edition: January 1983
Second Edition: January 1984
Latest Printing: April 1984

DISCLAIMER - CompuPro makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, CompuPro reserves the right to revise this publication and to make any changes from time to time in the content hereof without obligation of CompuPro to notify any person of such revision or changes.

CONTENTS

SYSTEM CONFIGURATION:

The following boards are required for booting the CompuPro FORTH system:

CPU 68K
64K bytes of CompuPro RAM
DISK1A or DISK1 floppy disk controller
INTERFACER 1,2,3, or 4

SWITCH SETTINGS AND JUMPER OPTIONS

The following list will describe the proper switch settings and jumper options required for each CompuPro board when running the CompuPro FORTH system.

CPU 68K -- All positions of all switches OFF.

Jumper J1	-	B-C Connected
Jumper J2	-	OFF (for 4 or 8 MHz operation)
		ON (for 5 or 10 MHz operation)
Jumper J3	-	OFF
Jumper J4	-	OFF
Jumper J5	-	INSTALLED
Jumper J6	-	INSTALLED
Jumper J7	-	ON
Jumper J8	-	ON
Jumper J9	-	A to J10 A installed
Jumper J10	-	A to J9 A installed
Jumper J11	-	don't care
Jumper J12	-	don't care
Jumper J13	-	don't care
Jumper J14	-	ON
Jumper J15	-	OFF
Jumper J16	-	B-C Connected
Jumper J17	-	don't care
Jumper J18	-	don't care

DISK 1A - The standard switch settings for running 8" floppies as drives "A" and "B", and 5.25" floppies as drives "C" and "D" are as follows:

"OFF"	S1	"ON"	"OFF"	S2	"ON"	"OFF"	S3	"ON"
	1	>	<	1		<	1	
	2	>	<	2			2	>
	3	>	<	3			3	>
*	4	*	<	4			4	>
*	5	*	<	5			5	>
*	6	*	<	6		<	6	
<	7		<	7		<	7	
<	8			8	>		8	>

*S1 positions 4-6 must be set as shown below depending on the type of CPU being used and the I/O device being used as the console.

S1 POSITION			CPU TYPE	CONSOLE I/O DEVICE
4	5	6		
ON	ON	ON	CPU 86/87	INTERFACER 1/2
ON	ON	OFF	CPU 68K	SYS. SUP./INTERFACER 3/4
ON	OFF	ON	CPU 86/87	SYSTEM SUPPORT
ON	OFF	OFF	CPU 86/87	INTERFACER 3/4
OFF	ON	ON	CPU 85/88 - Z	INTERFACER 1/2
OFF	ON	OFF	NOT SUPPORTED	
OFF	OFF	ON	CPU 85/88 - Z	SYSTEM SUPPORT
OFF	OFF	OFF	CPU 85/88 - Z	INTERFACER 3/4

J1 - POSITION "5"
J2 - POSITION "5"
J3 - POSITION "8"
J4 - POSITION "8"
J5 - REMOVED
J6 - A-C FOR MINIFLOPPIES GENERATING READY, OTHERWISE B-C.
J7 - B-C (TWO WAIT STATES)
J8 - LEAVE AS SHIPPED
J9 - LEAVE AS SHIPPED
J10- SHUNT ON "4"
J11- SHUNT INSTALLED

These settings select DMA arbiter priority 15, port C0H-C3H, wait states enabled, and the BOOT routine as selected.

DISK 1

"OFF"	S1	"ON"	"OFF"	S2	"ON"
	1	>		1	>
<	2		<	2	
<	3		<	3	
<	4		<	4	
<	5			5	>
<	6			6	>
<	7			7	>
<	8			8	>

Jumper J16 - B-C Connected
Jumper J17 - B-C Connected

The EPROM provided with the FORTH diskette replaces the existing EPROM in your DISK1 controller board. The only difference between your old EPROM and the new EPROM is that the "bit-banger" routine for the on-board serial port has been replaced with the 68000 boot routine for booting the FORTH system on the CPU 68K.

RAM 16 and RAM 21

"OFF"	S1	"ON"
	1	>
	2	>
	3	>
	4	>
	5	>
	6	>
	7	>
	8	>

RAM 17

"OFF"	S1	"ON"	"OFF"	S2	"ON"
<	1		<	1	
	2	>		2	>
	3	>		3	>
	4	>		4	>
	5	>		5	>
	6	>		6	>
	7	>		7	>
	8	>		8	>
	9	>		9	>
	10	>		10	>

INTERFACER 1

"OFF"	S1	"ON"	"OFF"	S2	"ON"	"OFF"	S3	"ON"
	1	>		1	>	<	1	
<	2			2	>		2	>
<	3			3	>		3	>
<	4		<	4			4	>
	5	>		5	>		5	>
<	6			6	>		6	>
<	7			7	>		7	>
<	8		<	8		<	8	

Terminal at ports 10 and 11 (hex) and printer at 2 & 3.
Terminal and printer are at 9600 baud.

INTERFACER 2

"OFF"	S2	"ON"	"OFF"	S3	"ON"
	1	>		1	>
<	2			2	>
<	3			3	>
<	4		<	4	
	5	>		5	>
<	6			6	>
<	7			7	>
<	8		<	8	

INTERFACER 3

"OFF"	S1	"ON"
<	1	
	2	>
	3	>
	4	>
<	5	
	6	>
	7	>
	8	>

This selects base port 10 for the Interfacer 3 with the terminal relative user 7 and the printer relative user 6.

Jumper J1 and J2 should be set in the SLAVE mode. The USART's are set for 8 data bits, no parity, asynch operation, RTS and DTR outputs in the "spacing" condition.

INTERFACER 4

"OFF"	S1	"ON"	"OFF"	S2	"ON"	"OFF"	S3	"ON"
<	1			1	>		1	>
<	2			2	>		2	>
<	3		<	3			3	>
<	4		<	4			4	>
<	5			5	>	<	5	
<	6			6	>	<	6	
<	7			7	>	<	7	
<	8		<	8		<	8	
<	9			9	>			
<	10		<	10				

This configuration sets the INTERFACER 4 at the base port 10 with the relative user 7 at 9600 baud and the printer relative user 6 at 9600 baud also. USART's are configured for 8 data bits, no parity, asynch operation, RTS and DTR outputs in the "spacing" condition. See note 3 below for CENTRONICS PARALLEL printer.

1. Jumper Sockets JS1-JS3 should contain 8 position DIP SHUNTS.
2. Jumper J6, J7, or J8 should be installed for 1, 2, or 3 waits.
3. For a serial printer, install J26 A-B and C-D. For a CENTRONICS PARALLEL printer, install J26 A-C and B-D.
4. All other jumpers may be removed.

Turn the system ON if it is OFF, and hit the reset button on the front panel. The light on drive 0 should begin blinking. Insert a system disk in drive 0 and close the door. The disk controller will read the system into memory and begin execution. The sign-on message "68000 mapForth 1.0" should appear. The system is now up and running. Pushing the Carriage Return (CR) key should produce the response "OK". When I tell you to type a command such as "4 5 + .", do not type the quotes (") and be sure to include at least one space between words. All command lines are ended by pushing the CR key. To load all of the normal extensions to the system, type "OK". The message "Loading system extensions ..." should appear. As each of the extensions (Editor, Assembler, &c.) is loaded more messages will appear. When loading is complete (it takes less than a minute) the message "System is loaded. Size = nnn " will appear, where nnn will be the number of bytes in the dictionary.

*** * * IMPORTANT NOTE: * * ***

The FIRST thing to do now is to make a copy (backup) of the system disk. This can help to prevent embarrassment, unpleasant delays, and extra cost. To back up the system disk, you will need a disk formatted in double density with 1024 bytes per sector, ideally by a Disk 1. If necessary, follow the instructions in the next paragraph to format the required disk. Make certain that the new disk is write-enabled (cover the write-protect notch if there is one). Type "COPYING LOAD DISK-COPY" and follow the prompts. Then remove your original system disk and put it in your archival vault. Move your new system disk to drive 0 and feel free to destroy it with impunity.

* * * * * * * * * * * * * * * * *

Formatting a disk

Perform the following steps: with a system disk in drive 0 type "FORMATTING LOAD". When the system responds with OK, you can remove the system disk to be extremely cautious. Type "FORMAT". In response to the questions "Drive?" and "Density?" type "1" and "3" respectively, to format a disk in drive 1 with density 3, which is 1024 bytes per sector. When formatting is finished the system will say OK. If you are backing up your system disk, then replace it in drive 0 and return to the previous paragraph.

Starting Forth

The book Starting Forth, by Leo Brodie, published by Prentice Hall, is required reading and a useful reference for this system. With very few exceptions, mapForth is compatible with the system described in Starting Forth. The differences are discussed later. In the few cases where Starting Forth and the 79-Standard differ,

preference was given to Starting Forth. This choice was made both to simplify the task of providing documentation, and because the implementation in Starting Forth was felt to be somewhat better. The 83-Standard is likely to be closer to Starting Forth.

FIG

The Forth Interest Group (FIG) has performed a great service to the Forth community by their work in promoting Forth and making it so readily available. Without their help, this system would not have been possible. The magazine Forth Dimensions, which is published by FIG, is a good source of information on programming techniques, the standardizaton effort, and other issues of current interest. It has grown steadily better for the past few years, and I recommend it highly. Membership in FIG includes Forth Dimensions and is $15.00 per year. A subscription can be obtained by sending $15.00 to FIG, P. O. Box 1105, San Carlos, CA 94070.

Overview

mapForth is a stand-alone operating system, a programming environment, and a compiler/interpreter. It consists of a precompiled portion and a number of extensions in source code form. The extensions include an editor, an assembler, a memory dump utility, a decompiler, and a CP/M\ file utility, among others. Drivers are provided for many CompuPro boards. The System Support 1, if present, will be used by the editor and printing utility to provide them with the current date and time.

Shadows

Associated with each screen of source code is another screen which contains only comments, called a shadow screen. When editing, use the A (for Alternate) command to switch from a screen to its shadow, or back. When shadowing is used, shadows will occupy the upper half of a disk of any format.

Assembler

A full 68000 assembler is provided. It allows defining Forth words in machine code, defining subroutines or interrupt handlers, and so forth. Motorola mnemonics are used for instructions. A full set of structured conditionals is provided, including DO LOOPs. Macro capability is naturally available through : (colon). The addressing modes use a fairly normal Forth assembler notation.

Editor

A screen oriented line editor is provided which is compatible with chapter 3 of Starting Forth. Some extensions have been added. It has a terminal independent part, and a continuous redisplay capability which is terminal dependent. To adapt the editor to your terminal, edit in the definitions of the four terminal control words using L to display the effect of the editing commands. The four terminal control words are AT, which performs direct cursor

positioning, DARK, which clears the screen and homes the cursor, BLOT, which clears to the end of the current line, and -LINE, which deletes the current line, causing those below to scroll upwards. Reload the system with these definitions in place of the defaults, then use " n EDIT " to edit screen number n.

Disk Interface

One of the more unusual (for Forth) features of this system is the way it handles the disk drives. Most disk parameters, such as the number of sectors per track, are kept in arrays with an entry for each drive. These arrays act exactly like variables, except that the address returned is a function of DRV, the current drive. Whenever a drive is selected (with DRIVE), the density, sector skewing, and all other parameters are automatically set to match the disk in the drive. This lets BLOCK work between disks which have very different formats with very little bother. In addition, a utility to access data in CP/M format files is provided, which supports all CompuPro formats. This also works on a per drive basis and is transparent to BLOCK. Once a disk is identified as containing CP/M files by the user, the system remembers and will access it accordingly.

Pre-compiled System

The pre-compiled portion of the system was generated by a meta-compiler, and placed on track one of a disk formatted with 1024 bytes per sector. When the system boots, the disk controller reads all of track one of such a disk into memory at address 400(hex) and jumps to the start of the code. The pre-compiled system has just enough capability to be able to extend itself from the disk. The word OK is used to extend the primitive system by loading screen 32, which loads all desired utilities. When you type the command SAVE-SYSTEM, the extended system will be written to disk. It will clobber the original system, so be sure to have a backup copy. The next time you boot, the portion of the system loaded by the disk controller will read in the binary image of the extensions, as well as the exception vectors, and you will be right back where you were when you saved the system.

Exception Handling

The 68000 has 1024 bytes reserved for "exception vectors", which include interrupts and error traps. All of the exception vectors are initialized to execute the WARM boot routine. After the assembler is loaded, some of the exception vectors such as Address Error are set to execute routines which print error messages.

This glossary gives descriptions of the words in release 1 of mapFORTH. The words are presented in order of their ASCII sort.

The first line of each entry describes the effect of execution on the parameter stack, if any:

before -- after

Two dashes "--" indicate the execution point. In this notation, the top of the stack is to the right.

Stack parameter abbreviations:

adr	memory address
b	8 bit byte (i.e. upper 8 bits are zero)
c	7 bit ASCII character (high 9 bits are zero)
d	32 bit signed double integer, most significant portion with sign on top of stack
f	boolean flag, zero is false, non-zero is true
false	boolean false flag, equals zero
n	16 bit signed integer number
u	16 bit unsigned integer
true	boolean true flag, equals 1 (or non-zero)

Attributes

The capital letters on the right indicate attributes of the defined words.

C	May only be used during compilation.
E	Intended for execution only.
I	Immediate. Will execute even when compiling.
U	A user variable.

Unless otherwise noted, all references to numbers are for 16 bit signed integers. For 32 bit numbers, the most significant part is on the top. All arithmetic is implicitly 16 bit signed integer math.

< null > I

The name of this word is a single ASCII null. Its function is to terminate text interpretation from the terminal or from a disk buffer, as both buffers always leave a null at the end.

! n adr --

Store 16 bits of n at address. ´store´.

!CSP

Save the stack position in CSP. Used as part of the compiler security.

!DRV abs-block -- rel-block

Sets the drive number in DRV based on the absolute block number given and the CAPACITYs of the drives. Returns the block number relative to the selected drive.

!DENSITY skew density --

Skew is the address of a sector skewing table. Density is a number from 0 to 3, equal to the log2 of bytes/sector divided by 128. !DENSITY sets SKEW, DENSITY, TRK/DRV, SEC/TRK, SEC/BLK, SEC/DRV, and CAPACITY.

" -- adr c (execution)
-- (compilation)

Used in the form:
" string"

Compiles an inline string. If compiling, it precedes the string with (") which will return the address and length of the string when the word is executed. If executing, it leaves the address and length of the string on the stack. ´quote´

d1 -- d2

Generate from a double number d1, the next ASCII character which is placed in an output string. Result d2 is the quotient after division by BASE, and is maintained for further processing. Used between <# and #>. See #S.

#> d -- adr count

Terminates numeric output conversion by dropping d, leaving the text address and character count suitable for TYPE.

#BUFFERS -- n

A constant returning the number of disk buffers allocated.

#DRIVES -- n

A constant returning the number of disk drives in the system.

#S d1 -- d2

Generates ASCII text in the text output buffer, by the use of #, until a zero double number n2 results. Used between <# and #>.

#TIMES -- adr

A variable containing the number of times the input text has been interpreted. Used by TIMES. ´number of times´

#WORDS -- n

Counts the number of words in the current dictionary search order. ´number of words´

$.ARRAY n -- (compilation)
m -- adr n (execution)

A defining word which creates arrays of strings of length n. When the array is executed, it returns the address and length of the m-th string.

´ -- adr

Used in the form: ´ <name>

Leaves the parameter field address of dictionary word <name>. If the word is not found after a search of CONTEXT and CURRENT, an appropriate error message is given. ´tick´

(I

Used in the form: (comment)

Ignore a comment that will be delimited by a right parentheses on the same line. May occur during execution or in a colon-definition. A blank after the leading parentheses is required.

(.") C

The runtime procedure, compiled by ." which transmits the following in-line text to the selected output device. See ."

(;CODE) C

The run-time procedure, compiled by ;CODE, that rewrites the code field of the most recently defined word to point to the following machine code sequence. See ;CODE.

(+LOOP) n -- C

The run-time procedure compiled by +LOOP, which increments the loop index by n and tests for loop completion. See +LOOP.

(DO) C

The run-time procedure compiled by DO which moves the loop-control parameters to the return stack. See DO.

(FIND) addr1 addr2 -- pfa b tf (ok)
addr1 addr2 -- ff (bad)

Searches the dictionary starting at the name field address addr2, matching to the text at addr1. Returns parameter field address, length byte of name field, and boolean true flag for a good match. If no match is found, only a boolean false flag is left on stack.

(LOOP) C

The run-time procedure compiled by LOOP which increments the loop index and checks for loop completion. See LOOP.

***** n1 n2 -- prod

Leave the signed product of two signed numbers.

***/** n1 n2 n3 -- n4

Leave the ratio n4 = n1 * n2 / n3 where all are signed numbers. Retention of an intermediate 31 bit product permits greater accuracy than would be possible with the sequence: n1 n2 * n3 /

***/MOD** n1 n2 n3 -- n4 n5

Leave the quotient n5 and remainder n4 of the operation: n1 * n2 / n3. An 31 bit intermediate product is used as for */.

+ n1 n2 -- sum
Leave the sum of n1+n2

+! n adr --
Add n to the value at the address. 'plus-store'.

+- n1 n2 -- n3
Apply the sign of n2 to n1, which is left as n3.

+LOAD n --
LOAD the n-th block following the current one, which is in BLK.

+LOOP n1 -- (execution)
adr n2-- (compilation) I,C

Used in a colon definition in the form: DO ... n1 +LOOP
At run-time, +LOOP selectively controls branching back to the corresponding DO based on n1, the loop index and the loop limit. The signed increment n1 is added to the index and the total compared to the limit. The branch back to DO occurs until the new index is equal to or greater than the limit (n1>0), or until the index is equal to or less than the limit (n1<0). Upon exiting the the loop, the parameters are discarded and execution continues ahead.

At compile-time, +LOOP compiles the run-time word (+LOOP) and the branch offset computed from HERE to the address left on the stack by DO. n2 is used for compile time error checking.

+THRU u1 u2 --
LOAD a range of blocks from u1 to u2 above the current, which is in BLK.

, n --
Store n into the next available dictionary memory cell, advancing the dictionary pointer. 'comma'.

- n1 n2 -- difference
Leave the difference of n1 - n2.

--> I
Continue interpretation with the next disc screen. 'next-screen'.

-FIND -- pfa b true (found)
-- false (not found)

Accepts the next text word (delimited by blanks) in the input stream to HERE, and searches the CONTEXT and then CURRENT vocabularies for a matching entry. If found, the dictionary entry's parameter field address, its length byte, and a boolean true flag is left on the stack. Otherwise, only a boolean false flag is left.

-LINE

A terminal dependent routine which deletes the line containing the terminal's cursor.

-ROT n1 n2 n3 -- n3 n1 n2

Rotates the top of three stack items below the other two.

-TEXT adr1 n adr2 -- r

Compares strings at adr1 and adr2 of length n. Returns +1 for string 1 > string 2, 0 for equal, -1 for string 1 < string 2.

-TRAILING adr n1 -- adr n2

Adjusts the character count n1 of a text string beginning at adr to suppress the output of trailing blanks, i.e. the characters at addr+n1 to addr+n2 are blanks.

. n --

Print a number from a signed 16 bit two's complement value, converted according to the numeric BASE. A trailing blank follows. 'dot'.

." I

Used in the form: ." string"

Compiles an in-line string string (delimited by a trailing ") with an execution procedure to transmit the text to the selected output device. If executed outside a definition, ." will immediately print the string. See (.") .

.R n1 n2 --

Print the number n1 right aligned in a field whose length is n2. No following blank is printed.

.S

Prints the contents of the stack non-destructively. The top of stack is to the right, and values are shown unsigned.

.STATUS

An execution vector. Default value is CR. Used by QUIT. If your terminal has a status line, set .STATUS to print the status of the system on it.

/ nl n2 -- quot

Leave the signed quotient of nl/n2.

/DRIVE

A defining word which creates arrays of disk variables. Words created by /DRIVE act like variables, but the address returned is a function of DRV, the current drive. Words defined by /DRIVE include SEC/TRK, SEC/DRV, TRK.DRV, CAPACITY, DS?, and CPM?.

/LOOP nl -- (run)
adr n2-- (compile) I,C

Used in a colon definition in the form: DO ... nl /LOOP At run-time, /LOOP selectively controls branching back to the corresponding DO based on nl, the loop index and the loop limit. The unsigned increment nl is added to the index and the total compared to the limit. The branch back to DO occurs until the new index is equal to or greater than the limit. Upon exiting the loop, the parameters are discarded and execution continues ahead.

At compile-time, /LOOP compiles the run-time word (/LOOP) and the branch offset computed from HERE to the address left on the stack by DO. n2 is used for compile time error checking.

/MOD nl n2 -- remainder quotient

Leave the remainder and signed quotient of nl/n2. The remainder has the sign of the dividend.

0 1 2 3 -- n

These small numbers are used so often that it is attractive to define them by name in the dictionary as constants.

0< n -- f

Leave a true flag if the number is less than zero (negative), otherwise leave a false flag.

0= n -- f

Leave a true flag if the number is equal to zero, otherwise leave a false flag.

0> n -- f
Leave a true flag if the number is greater than zero, otherwise leave a false flag.

1+ n1 -- n2
Increment n1 by 1.

1- n1 -- n2
Decrement n1 by 1.

1SHADOW -- n
Returns the block number of the first shadow screen on the current drive.

2! nlow nhigh adr --
32 bit store. nhigh is stored at addr, nlow is stored at addr+2.

2* n1 -- n2
n2 is double n1.

2+ n1 -- n2
Leave n1 incremented by 2.

2, d --
Appends a 32 bit number to the dictionary.

2- n1 -- n2
Leave n1 decremented by 2.

2/ n1 -- n2
n2 is the arithmetic right shift of n1.

2@ adr -- nlow nhigh
32 bit fetch. nhigh is fetched from addr, nlow is fetched from addr+2.

2DROP n2 n1 --
Drops the two top values from the stack.

2DUP n2 n1 -- n2 n1 n2 n1
Duplicates the top two values on the stack. Equivalent to OVER OVER.

2OVER d1 d2 -- d1 d2 d1

Places a copy of the second 32 bit value on the stack onto the stack.

2SWAP d1 d2 -- d2 d1

Exchanges the top two double numbers on the stack.

3DROP n3 n2 n1 --

Drops the three top values from the stack.

3DUP n3 n2 n1 -- n3 n2 n1 n3 n2 n1

Duplicates the top three values on the stack.

4DUP n4 n3 n2 n1 -- n4 n3 n2 n1 n4 n3 n2 n1

Duplicates the top four values on the stack.

: I,E

Used in the form called a colon-definition:

: <name> ... ;

Creates a dictionary entry defining <name> as equivalent to the following sequence of Forth word definitions '...' until the next ';' or ';CODE'. The compiling process is done by the next interpreter as long as STATE is non-zero. Other details are that the CONTEXT vocabulary is set to the CURRENT vocabulary and that words with the precedence bit set are executed rather than being compiled.

::

Compiles a nameless word (an orphan), executes it, and forgets it. Allows executing control structures at the terminal. This is referred to as immediate compilation.

; I,C

Terminate a colon-definition and stop further compilation. Compiles the run-time EXIT.

;CODE I,C

Used in the form:

: <name1> xxx ;CODE yyy C;

where xxx is high level code and yyy is assembler mnemonics. Stop compilation and terminate a new defining word <name1> by compiling (;CODE). Set the CONTEXT vocabulary to ASSEMBLER, assembling to machine code the following mnemonics. When <name1> later executes in the form:

<name1> <name2>

the word <name2> will be created with its execution

procedure given by the machine code following <name1>. That is, when <name2> is executed, it does so by jumping to the code after <name1>. An existing defining word must exist in <name1> prior to ;CODE.

< n1 n2 — f

Leave a true flag if n1 less than n2, otherwise leave false flag.

<#

Setup for pictured numeric output formatting using the words:

<# # #S SIGN #>

The conversion is done on a double number producing text at PAD.

<> n1 n2 — f

Leave a true flag if n1 does not equal n2, else leave a false flag.

<CMOVE adr1 adr2 n

Move n bytes from adr1 to adr2, moving the bytes at the high addresses first. ´reverse-cmove´

= n1 n2 — f

Leave a true flag if n1=n2, otherwise leave a false flag.

=< n1 n2 -- f

Leave a true flag if n1 is less than or equal to n2, else leave a false flag.

> n1 n2 — f

Leave a true flag if n1 is greater than n2, otherwise leave a false flag.

>= n1 n2 -- f

Leave a true flag if n1 is greater than or equal to n2, else leave a false flag.

>B n -- lb hb

Split a number into bytes, leaving high byte on top.

>BUFFERS -- adr

Return the address of the disk buffer assignment table. The entries consist of a block number (-1 if not allocated), a buffer address, and an update flag.

>HI

An ASSEMBLER macro which compiles an exit to high level. Allows executing high level code from assembler routines.

>IN -- adr U

A user variable containing the byte offset within the current input text buffer (terminal or disk) from which the next text will be accepted. WORD uses and alters the value of >IN.

>OUT -- adr U

A user variable that contains a value incremented by EMIT. The user may alter and examine >OUT to control display formatting.

>R n --

Remove a number from the computation stack and place as the most accessable on the return stack. Use should be balanced with R> in the same definition.

>TYPE adr n --

Same as TYPE, but moves string to PAD before typing.

>UPC c1 -- c2

Converts lower case characters to upper case. All others are unaffected.

>W lb hb -- n

Combines two bytes into a 16 bit number ('word').

? adr --

Print the value contained at the address in free format according to the current base.

?2SIDE

Sets DS? for the current drive to 0 if single sided, or to 8 if double sided.

?BRANCH f -- C

The run-time procedure to conditionally branch. If f is false (zero), the following in-line parameter is added to the interpretive pointer to branch ahead or back. Compiled by IF, UNTIL, and WHILE.

?COMP

Issue error message if not compiling.

?CR

Executes CR if >OUT is greater than C/L.

?CSP

Issue error message if stack position differs from that saved in CSP.

?DO -- adr n

Used in the form:

?DO ... LOOP or ?DO ... +LOOP

Compiler directive similar to DO which compiles (?DO) and a branch offset. At execution time, if the parameters passed to the loop are equal, the loop is executed zero times.

?DUP n1 -- n1 (if zero)
n1 -- n1 n1 (if non-zero)

Reproduce n1 only if it is non-zero. This is usually used to copy a value just before IF, to eliminate the need for an ELSE part to drop it.

?ENOUGH n --

Prints error message and exits if number of parameters on the stack is less than n.

?EXEC

Issue an error message if not executing.

?INPUT -- n true
-- false

Converts input text into a number, and returns the value and a true flag. If no text is available, returns false.

?KEY -- f

True flag indicates key available.

?LOADING

Issue an error message if not loading.

?MISSING f --

If flag is true, prints the name at HERE followed by ? then aborts.

?OK -- f

Returns true if disk status is ok. If drive was not ready or write protected, prints message and aborts. Other errors return false flag.

?PAIRS n1 n2 --

Issue an error message if n1 does not equal n2. The message indicates that compiled conditionals do not match.

?R/W f --

Primitive used by (R/W) to perform disk reads and writes. True flag causes read; false flag causes write.

?STACK

Issue an error message if the stack is out of bounds.

@ adr -- n

Leave the 16 bit contents of address.

A

Changes SCR from a screen to its shadow, or vice versa.

ABS n -- u

Leave the absolute value of n as u.

ABORT" -- (compilation)
f -- (execution)

Used in a colon definition in the form:

ABORT" string"

Compiles (ABORT") followed by the string. When the colon definition is executed, if the flag is true the string is typed and an abort is performed. If the flag is false, no action is taken.

ADDRESS -- adr

A variable containing the address in memory of the data transferred to or from the disk. EXT-ADDRESS contains the high 8 bits, ordinarily zero.

AGAIN adr n -- (compiling) I,C

Used in a colon-definition in the form:

```
BEGIN ... AGAIN
```

At run-time, AGAIN forces execution to return to the corresponding BEGIN. There is no effect on the stack. Execution cannot leave this loop (unless R> DROP is executed one level below).

At compile time, AGAIN compiles BRANCH with an offset from HERE to addr. n is used for compile time checking.

ALIGN

Used to keep the dictionary pointer on an even address. If DP contains an odd address, ALIGN appends a byte containing a blank to the dictionary.

ALLOT n --

Add n to the dictionary pointer DP. May be used to reserve dictionary space.

AND n1 n2 -- n3

Leave the bitwise logical and of n1 and n2 as n3.

ASCII -- c I

Used in the form:

```
ASCII q
```

Returns the value of the following character. If compiling, the value is compiled as a literal.

ASSEMBLER I

A vocabulary containing the 68000's assembler mnemonics and related words. Invoked by CODE and ;CODE.

ASSEMBLING -- n

A constant returning the starting screen number of the file containing the source code for the ASSEMBLER.

AT col row --

A terminal dependent routine which positions the cursor.

B

Decrements SCR by one. 'back'

B/BUF -- n

This constant leaves the number of bytes per disc buffer, the byte count read from disc by BLOCK.

B/SEC -- adr

An array of variables containing the number of bytes per sector for each drive. The address returned is a function of DRV.

BACK adr --

Calculate the backwards branch offset from HERE to adr and compile into the next available dictionary memory address. A primitive used by the structured conditionals.

BACK-UP n -- 0

A primitive used by EXPECT to cancel the current line upon receipt of a ^X.

BASE -- adr U

A user variable containing the current number base used for input and output conversion.

BEGIN -- adr n (compiling) I,C

Occurs in a colon-definition in the form:

```
BEGIN   ...   UNTIL
BEGIN   ...   AGAIN
BEGIN   ...   WHILE   ...   REPEAT
```

At run-time, BEGIN marks the start of a sequence that may be repetitively executed. It serves as a return point from the corresponding UNTIL, AGAIN, or REPEAT. When executing UNTIL, a return to BEGIN will occur if the top of the stack is false; for AGAIN and REPEAT a return to BEGIN always occurs.

At compile time BEGIN leaves its return address and n for compiler error checking.

BINARY

Sets BASE to 2.

BL -- c

A constant that leaves the ASCII value for 'blank'.

BLANK adr count --

Fill an area of memory beginning at adr with blanks.

BLK -- adr U

A user variable containing the block number being interpreted. If zero, input is being taken from the terminal input buffer.

BLOCK n -- adr

Leave the memory address of the block buffer containing block n. If the block is not already in memory, it is transferred from disc to whichever buffer was least recently written. If the block occupying that buffer has been marked as updated, it is rewritten to disc before block n is read into the buffer. See also BUFFER, R/W, UPDATE, FLUSH.

BLOT col --

A terminal dependent routine to clear the terminal screen from the cursor to the end of the line. Cursor is at col.

BOOT

An execution vector used for special cold start initialization or turnkey applications. It is executed in COLD just before QUIT.

BRANCH C

The run-time procedure to unconditionally branch. An in-line offset is added to the interpretive pointer IP to branch ahead or back. BRANCH is compiled by ELSE, AGAIN, REPEAT.

BS-IN n1 -- n2

A primitive routine used by EXPECT to handle back-space and delete characters. n1 is decremented unless already zero, in which case the bell is sounded.

C! b adr --

Store 8 bits of b at address.

C, b adr --

Store 8 bits of b into the next available dictionary byte, advancing the dictionary pointer.

C/L -- n

Constant leaving the number of characters per line, used by Editor.

C@ adr -- b

Leave the 8 bit contents of memory address on the stack.

CAPACITY -- adr

An array of variables containing the capacity in blocks of each disk drive. adr is a function of DRV.

CASE -- adr

A variable containing a flag indicating whether SEARCH will ignore upper/lower case of characters. False means ignore case.

CASE: n --

A defining word which creates positional case statements. Used in the decompiler.

CC -- adr

A table of routines to be executed for each of the control characters. Used by EXPECT.

CFA pfa -- cfa

Convert the parameter field address of a definition to its code field address.

CHAR n c -- n+1

A primitive routine used by EXPECT to handle normal characters.

CHOOSE n1 -- n2

n2 is a random number between 0 and n1-1.

CLOCK? -- f

Returns true if a System Support 1 with real time clock exists. False otherwise.

CMDBUF -- adr

The disk command buffer. Read and write commands are assembled here before being sent to the controller.

CMOVE from to count --

Move the specified quantity of bytes beginning at address from to address to. The contents of address from is moved first proceeding toward high memory.

CODE

A defining word used to create words which execute assembler routines. Used in the form:

```
CODE DUP   SP ) SP -) MOVE   NEXT  C;
```

COLD

The cold start procedure.

COMMAND

Sends nine bytes from CMDBUF to the disk controller.

COMPILE C

When the word containing COMPILE executes, the execution address of the word following COMPILE is copied (compiled) into the dictionary. This allows specific compilation situations to be handled in addition to simply compiling an execution address (which the interpreter already does).

CONINIT

Initializes the console UART. Sets Interfacer III or IV to 9600 baud. Does not affect Interfacer I or II.

CONSOLE -- n

The I/O port address of the console. Usually 10(hex).

CONSTANT n --

A defining word used in the form:

n CONSTANT <name>

to create word <name>, with its parameter field containing n. When <name> is later executed, it will push the value n onto the stack.

CONSTATUS -- n

Reads the status byte from the console UART.

CONTEXT -- adr U

A user variable containing a pointer to the vocabulary within which dictionary searches will first begin.

CONVERT d1 addr1 -- d2 addr2

Convert the ASCII text beginning at addr1+1 with regard to BASE. The new value is accumulated into double number d1, being left as d2. Addr2 is the address of the first unconvertable digit. Used by NUMBER.

CONVEY first last --

Moves a block of screens. Usage:

first last TO dest CONVEY

first last TO dest ON1 CONVEY

The word TO computes an offset and stores it in SKIPPED. FIRST and LAST are the first and last source screens to move, DEST is the first destination screen.

COPY u1 u2 --

COPY screen u1 onto screen u2.

COUNT addr1 -- addr2 n

Leave the byte address addr2 and byte count n of a message text beginning at address addr1. It is presumed that the first byte at addr1 contains the text byte count and the actual text starts with the second byte. Typically COUNT is followed by TYPE.

CPM? -- adr

An array of variables indicating whether each drive contains Forth blocks or CP/M-style files.

CR

An execution vector; default value is CRLF.

CR-IN n1 adr n2 -- n1 adr n1

A primitive routine used by EXPECT to handle Carriage Return input, which terminates an input line.

CRASH

Prints a message and aborts. Used to initialize execution vectors.

CRLF

Transmits a carriage return and line feed to the selected output device.

CREATE

A defining word used in the form:

CREATE <name>

by such words as CODE and CONSTANT to create a dictionary header for a FORTH definition. The code field contains the address of the code for variables. The new word is created in the CURRENT vocabulary.

CSP -- adr U

A user variable temporarily storing the stack pointer position, for compilation error checking.

CURRENT -- addr

A user variable containing a pointer to the vocabulary to which new definitions will be added. See DEFINITIONS.

D+ d1 d2 -- dsum

Leave the double number sum of two double numbers.

D+- d1 n -- d2

Apply the sign of n to the double number d1, leaving it as d2.

D- d1 d2 -- difference

Leave the double precision difference of two double numbers.

D. d --

Print a signed double number from a 32 bit two's complement value. The high-order 16 bits are most accessible on the stack. Conversion is performed according to the current base. A blank follows. "d dot".

D.R d n --

Print a signed double number d right aligned in a field n characters wide.

DABS d -- ud

Leave the absolute value ud of a double number.

DARK

A terminal dependent clear screen routine.

DATA -- p

The I/O port number for the DISK 1 data/command port.

DATE

Prints the date.

DAY n --

Returns the address and length of a string representing the name of the n-th day of the week. 0 is Sunday, etc.

DECIMAL

Set the numeric conversion BASE for decimal input-output.

DEFERRED

A defining word which creates execution vectors. Used in the form: DEFERRED LOAD
' (LOAD) IS LOAD

DEFINITIONS
Set the CURRENT vocabulary to the CONTEXT vocabulary.

DEFS
A short form of DEFINITIONS.

DENSE -- adr
A table containing several parameters which vary with density: SEC/BLK,GPL,DTL,SEC/TRK. DENSE is used by !DENSITY.

DENSITY -- adr
A variable used by the disk interface.

DEPTH -- n
Returns the number of values on the parameter stack.

DEX
A short name for DECIMAL.

DH adr n --
Display n bytes from memory starting at adr.

DI
Disable Interrupts.

DIGIT c n1 -- n2 true (ok)
c n1 -- false (bad)
Converts the ASCII character c (using base n1) to its binary equivalent n2, accompanied by a true flag. If the conversion is invalid, leaves only a false flag.

DISK-ERROR -- addr
A variable used by the disk interface, containing the disk status for the last sector read or written. 0 means no error detected.

DISPLACEMENT -- n
The offset between a screen and its shadow on the current drive.

DL n --
Dump line n from the current screen (SCR).

DLN adr —

Dump 16 bytes from memory at adr.

DLITERAL d -- d (executing) I
d -- (compiling)

If compiling, compile a stack double number into a literal. Later execution of the definition containing the literal will push it to the stack. If executing, the number will remain on the stack.

DNEGATE d1 -- d2

Convert double number d1 to its double number two's complement.

DMA -- p

The I/O port number for the DISK 1 DMA port.

DO n1 n2 -- (execute) I,C
adr n -- (compile)

Occurs in a colon-definition in the form:

```
DO  ...  LOOP
DO  ...  +LOOP
```

At run time, DO begins a sequence with repetitive execution controlled by a loop limit n1 and an index with an initial value n2. DO removes these from the stack. Upon reaching LOOP the index is incremented by one. Until the new index equals or exceeds the limit, execution loops back to just after DO; otherwise the loop parameters are discarded and execution continues ahead. Both n1 and n2 are determined at run-time and may be the result of other operations. Within a loop, "I" will copy the current value of the index to the stack. See: I, LOOP, +LOOP, LEAVE. When compiling within the colon definition, DO compiles (DO), leaves the following address adr and n for later error checking.

DOES>

A word which defines the run-time action within a high-level defining word. DOES> alters the code field of the new word to execute the sequence of compiled word addresses following DOES>. Usually used with CREATE in a colon definition. When the DOES> part executes it begins with the address of the first parameter of the new word on the stack.

DP -- adr U

A user variable, the dictionary pointer, which contains the address of the next free memory above the dictionary. The value may be read by HERE and altered by ALLOT.

DPL -- adr U,

A user variable containing the number of digits to the right of the decimal on double integer input. It may also be used to hold output column location of a decimal point, in user generated formatting. The default value on single number input is -1.

DS? -- adr

An array of variables containing a flag for each drive which indicates whether it is double-sided. adr is a function of DRV.

DRIVE n --

Selects drive n, automatically setting density and all related variables.

DROP n --

Drop the number from the stack.

DRV -- adr

A variable containing the most recently accessed disk drive.

DU adr -- adr+64

Dump 64 bytes from memory starting at adr.

DUMP adr n --

Print the contents of n memory locations beginning at addr. Both addresses and contents are shown in the current numeric base.

DUP n -- n n

Duplicate the value on the stack.

ED

Edit the current screen (SCR).

EDIT n --

Edit screen n, making it current.

EDITOR

A vocabulary containing the editing commands. Invoked by EDIT.

EI

Enable Interrupts.

ELSE addr1 n1 -- addr2 n2 (compiling) I,C

Occurs within a colon-definition in the form:

IF ... ELSE ... THEN

At run-time, ELSE executes after the true part following IF. ELSE forces execution to skip over the following false part and resumes execution following the THEN. It has no stack effect. At compile-time, ELSE emplaces BRANCH reserving a branch offset, leaves the address addr2 and n2 for error testing. ELSE also resolves the pending forward branch from IF by calculating the offset from addr1 to HERE and storing at addr1.

EMIT c --

Transmit ASCII character c to the selected output device. OUT is incremented for each character transmitted.

EMIT. c --

EMIT the character, substituting dot for any control character.

EMPTY

Forget all words defined after EMPTY.

EMPTY-BUFFERS

Mark all block-buffers as empty, not necessarily affecting the contents. Updated blocks are not written to the disk. This is also an initialization procedure before first use of the disk.

ENCLOSE addr1 c -- ddr1 n1 n2 n3

The text scanning primitive used by WORD. From the text address addr1 and an ASCII delimiting character c, is determined by the byte offset to the first non-delimiter character n1, the offset to the first delimiter after the text n2, and the offset to the first character not included n3. This procedure will not process past an ASCII null, treating it as an unconditional delimiter.

EPRINT -- adr

A variable containing a flag. If true, the printer is enabled.

ERASE adr n --

Clear a region of memory to zero from adr over n addresses.

ESC-IN

A primitive routine used by EXPECT to handle escape characters.

EVEN n1 -- n2

n2 equals n1 or n1+1, whichever is even.

EXCEPTS

Initializes all exception vectors to WARM start. This is the default value of the vector BOOT. It is not needed after a the extended system has been saved on disk by SAVE-SYSTEM, as the exception vectors are also saved.

EXCEPTIONS

A vocabulary containing the exception vector routines.

EXECUTE adr --

Execute the definition whose parameter field address is on the stack.

EXIT

Stop interpretation of a screen. EXIT is also the run-time word compiled at the end of a colon definition which returns execution to the calling procedure.

EXPECT adr count --

Transfer characters from the terminal t address, until a "return" or the count of characters has been received. Two nulls are added at the end of the text.

EXT-ADDRESS -- adr

A variable containing the 32 bit disk DMA address. ADDRESS points to the low half of EXT-ADDRESS.

EXTERNAL -- here E

Used to hide internal routines in the form:

```
INTERNAL ... EXTERNAL ... MODULE
```

Words defined between EXTERNAL and MODULE will remain findable, words defined between INTERNAL and EXTERNAL will be hidden.

FALSE -- false

A constant returning a false flag.

FENCE -- adr U

A user variable containing an address below which FORGETting is trapped. To forget below this point the user must alter the contents of FENCE.

FILL adr quan b --

Fill memory at the address with the specified quantity of bytes b.

FIRST -- n

A constant that leaves the address of the first (lowest) block buffer.

FLIP n1 -- n2

n2 is the one's complement of n1.

FLUSH

Write all updated disk buffers to the disk. Should be used after editing, before dismounting a disk, or before exiting FORTH.

FORGET

Executed in the form:

FORGET <name>

Deletes definition named <name> from the dictionary with all entries physically following it.

FORTH I

The name of the primary vocabulary. Execution makes FORTH the CONTEXT vocabulary. Until additional user vocabularies are defined, new user definitions become a part of FORTH. FORTH is immediate, so it will execute during the creation of a colon definition, to select this vocabulary at compile-time.

GET-SYS

Executed by COLD to bring any saved extensions in from disk. See BOOT and SAVE-SYSTEM.

H -- adr

A synonym for DP.

HERE -- adr

Leave the address of the next available dictionary location.

HEX

Set the numeric conversion base to sixteen (hexadecimal).

HI>

Used to return from high level code to assembler in a CODE definition. Used in the form:

```
CODE TEST  >HI  DUP  DROP  HI>  NEXT C;
```

HLD -- adr

A user variable that holds the address of the latest character of text during numeric output conversion.

HOLD c --

Used between <# and #> to insert an ASCII character into a pictured numeric output string; e.g. 2E HOLD will place a decimal point.

HOME

Restore the current disk to track zero.

I -- n C

Used within a DO ... LOOP to copy the loop index to the stack.

I' -- n C

Used within a DO ... LOOP to copy the loop limit to the stack.

ID. nfa --

Print a definition's name from its name field address.

IF f -- (run-time) I,C

-- adr n (compile)

Occurs in a colon-definition in the form:

```
IF ... THEN
IF ... ELSE ... THEN
```

At run-time, IF selects execution based on a boolean flag. If f is true (non-zero), execution continues ahead through the true part. If f is false (zero), execution skips till just after ELSE to execute the false part. After either part, execution resumes with the code after THEN. ELSE and its false part are optional; if missing, false execution skips to just after THEN. At compile-time IF compiles ?BRANCH and reserves space for an offset at addr. Addr and n are used later for resolution of the offset and error testing.

IMMEDIATE

Mark the most recently made definition so that when encountered at compile-time, it will be executed rather than compiled; i.e. the precedence bit in its header is set. This method allows definitions to handle unusual compiling situations, rather than build them into the fundamental compiler. The user may force compilation of an immediate definition by preceding it with [COMPILE].

INDEX start --

Print the first line of each screen from start until a key is struck. This is used to view the comment lines of an area of text on disk screens.

INPUT -- n

Accept and convert a number from the input text stream.

INTERNAL -- latest

Marks the start of a module. See EXTERNAL.

INTERPRET

The outer text interpreter which sequentially executes or compiles text from the input stream (terminal or disk) depending on STATE. If the word name cannot be found after a search of CONTEXT and then CURRENT it is converted to a number according to the current base. That also failing, an error message echoing the name with an "?" will be given. Text input will be taken according to the convention of WORD. If a decimal point is found as part of a number, a double number value will be left. The decimal point has no other purpose except to force this action. See NUMBER.

IS pfa --

Used in the form:

' (LOAD) IS LOAD

to set an execution vector. In the example, LOAD is set to execute (LOAD). If compiling, IS compiles (IS) and the pfa's of both the vector and its new value.

J -- n C

Used within nested DO ... LOOP's to copy the loop index of the next outer loop to the stack.

KEY -- c

An execution vector which leaves the ASCII value of the next terminal key struck. The default is (KEY).

L

List the current screen (SCR). See LIST

LABEL

A defining word which creates assembler labels.

LATEST -- addr

Leave the name field address of the topmost word in the CURRENT vocabulary.

LC! c d --

Store the low 8 bits of c at the 32 bit address d.

LC@ d -- c

Fetch a byte from the 32 bit address d.

LEAVE C

Force termination of a DO...LOOP at the next opportunity by setting the loop limit equal to the current value of the index. The index itself remains unchanged, and execution proceeds normally until LOOP or +LOOP is encountered.

LFA pfa -- lfa

Convert the parameter field address of a dictionary definition to its link field address.

LIMIT — n

A constant leaving the address just above the highest memory available for a disk buffer. This is the highest system memory.

LIST n --

Display the ASCII text of screen n on the selected output device. SCR contains the screen number during and after this procedure.

LIT — n C2,

Within a colon-definition, LIT is automatically compiled before each 16 bit literal number encountered in the input text. Later execution of LIT causes the contents of the next dictionary address to be pushed to the stack.

LITERAL n -- (compiling) I,C

If compiling, then compile the stack value n as a 16 bit literal. This definition is immediate so that it will

execute during a colon- definition. The intended use is:

```
: xxx [ calculate ] LITERAL ;
```

Compilation is suspended for the compile-time calculation of a value. Compilation is resumed and LITERAL compiles this value.

LOAD n --

Begin interpretation of screen n. LOADing will terminate at the end of the screen or at EXIT. See EXIT and --> .

LOCATE

A synonym for VIEW.

LOOP adr n -- (compiling) I,C

Occurs in a colon-definition in the form:

```
DO ... LOOP
```

At run-time, LOOP selectively controls branching back to the corresponding DO based on the loop index and limit. The loop index is incremented by one and compared to the limit. The branch back to DO occurs until the index equals or exceeds the limit; at that time, the parameters are discarded and execution continues ahead. At compile-time, LOOP compiles (LOOP) and uses adr to calculate an offset to DO. n is used for error-testing.

M* n1 n2 -- d

A mixed magnitude math operation which leaves the double number signed product of two signed numbers.

M/ d n1 -- n2 n3

A mixed magnitude math operator which leaves the signed remainder n2 and signed quotient n3, from a double number dividend and divisor n1. The remainder takes its sign from the dividend.

M/MOD ud1 u2 -- u3 ud4

An unsigned mixed magnitude math operation which leaves a double quotient ud4 and remainder u3, from a double dividend ud1 and single divisor u2.

MAX n1 n2 -- max

Leave the larger of two numbers.

MANY

Reinterpret the Text Input Buffer up to MANY until a key is struck. Used in the form:

```
xxx MANY
```

MIN n1 n2 — min
Leave the smaller of two numbers.

MISSING
Called by BLOCK if the desired block is not in memory. Assigns the least recently used buffer to the desired block, flush its contents to disk first if updated. Reads the new block into the buffer.

MOD n1 n2 — mod
Leave the remainder of n1/n2, with the same sign as n1.

MODULE pfa here' —
Terminates a module. See EXTERNAL.

MONTH n —
A string array which returns the address and length of a string representing the n-th month. 0 is Jan.

MOVE addr1 addr2 n --
Move n bytes beginning at addr1 into n cells beginning at addr2. This is a smart CMOVE.

MTB
A synonym for EMPTY-BUFFERS.

MYSELF
The recursion word. Compiles a reference to the word being defined.

N
Increments SCR. 'next'

NEGATE n1 — n2
Leave the two's complement of a number.

NEXT
The Forth address-interpreter. All code routines return to NEXT.

NFA pfa -- nfa
Convert the parameter field address of a definition to its name field.

NOOP

A FORTH "no-operation".

NOT f1 -- f2

f2 is the reverse of f1.

NUMBER adr -- d

Convert a character string left at adr with a preceding count, to a signed double number, using the current numeric base. If a decimal point is encountered in the text, its position will be given in DPL, but no other effect occurs. If numeric conversion is not possible, an error message will be given.

OCTAL

Sets BASE to 8.

OFFSET -- adr U

A variable which contains a block offset to disk drives. The contents of OFFSET is added to the stack number by BLOCK.

OK

Load screen 32. Used to begin extending a virgin system.

ON1

See CONVEY.

OR n1 n2 -- or

Leave the bit-wise logical or (product) of two 16 bit values.

OVER n1 n2 -- n1 n2 n1

Copy the second stack value, placing it as the new top.

P! b port# --

S-100 bus I/O port store. Outputs byte b to port#.

P@ port# -- b

S-100 bus I/O port fetch. Gets byte b from port#.

P-IN

Toggles EPRINT, the printer enable flag, and echoes a blank. A primitive routine used by EXPECT.

PAD -- adr

Leave the address of the text output buffer, which is a fixed offset above HERE.

PICK n1 -- n2

n2 is a copy of the n1-th stack value. 2 PICK is the same as OVER.

PFA nfa -- pfa

Convert the name field address of a compiled definition to its parameter field address.

QUERY

Input up to 80 characters of text (or until a "return") from the terminal. The text is placed at the address contained in TIB with >IN set to zero.

QUIT

Clear the return stack, stop compilation, and return control to the operator's terminal. No message is given.

R# -- adr U

A user variable which may contain the location of an editing cursor, or other file related function.

R/W adr blk f --

Read or write the disk. adr specifies the source or destination block buffer, blk is the block number, and f is a flag such that f=0 causes a disk write and f=1 causes a disk read. R/W is an execution vector whose default value is (R/W).

R0 -- adr U

A user variable containing the initial value of the return stack. "R zero". See RP!.

R> -- n

Remove the top value from the return stack and leave it on the computation stack. See >R and R@.

R@ -- n

Place a copy of the top item on the return stack onto the parameter stack. 'r-fetch'

RANDOM -- n

n is a random number. Uses RND.

RANGE adr n -- adr+n adr

Sets up a DO .. LOOP for a range of addresses.

REPEAT adr n -- (compiling) I,C

Used within a colon-definition in the form:

BEGIN ... WHILE ... REPEAT

At run-time, REPEAT forces an unconditional branch back to just after the corresponding BEGIN. At compile time, REPEAT compiles BRANCH and the offset from HERE to addr. n is used for error checking.

RESULTS

Reads status bytes from the disk controller to sequential addresses in STATBUF until all have been read.

RND -- adr

A variable containing the seed for the random number generator.

ROLL n --

The n-th stack item is first removed then transfered to the top of the stack, moving the remaining values into the vacated position. 3 ROLL is the same as ROT.

ROT n1 n2 n3 -- n2 n3 n1

Rotate the top three values on the stack, bringing the third to the top.

RP!

Initialize the return stack pointer from R0.

RP@ -- addr

Returns the contents of the return stack pointer register.

S"

Used in the form:

S" string"

Compiles the string into the dictionary, preceded by a count byte.

S+ adr n1 n2 -- adr+n2 n1-n2

Indexes into a string, returning the advanced address and reduced length.

S->D n -- d
Sign extend a single number to form a double number.

S-CONVEY
Moves a range of screens and their shadows. See CONVEY.

S0 -- adr U
A user variable that contains the initial value for the stack-pointer. "S zero". See SP!.

SAVE-BUFFERS
All blocks which have been UPDATEd are written to disk. The buffers remain assigned.

SAVE-SYSTEM
Places a bootable image of the currently running system onto the current drive. Places the size of the system into the variable SYS-SIZE.

SCR -- adr U
A user variable containing the screen number most recently referenced by LIST.

SDS -- n
Issues a sense drive status command and returns the result.

SEARCH adr1 n1 adr2 n2 -- n3 f
Search for string at adr1 of length n1 in buffer at adr2 of length n2. If found, return offset to start and true flag. Otherwise return 0 and false flag.

SEC/BLK -- adr
An array containing the number of sectors per block for each drive. When executed, returns the address of the value for the current drive (DRV).

SEC/TRK -- adr
An array containing the number of sectors per track for each drive. When executed, returns the address of the value for the current drive (DRV).

SECTOR -- adr
A variable containing the disk sector last accessed.

SEE

Used in the form:

SEE <name>

Prints a decompilation of the definition of the word <name>.

SEEK n --

Performs a seek to track n on the current disk drive. Sets TRACK to n.

SELECT -- n

The port number of the Interfacer IV user selection port. Set to 17(hex).

SEND n --

Send a byte to the disk controller.

SET-CMD

Sets the disk command buffer CMDBUF for the next command using TRACK, SECTOR, DRV, SIDE, and DENSITY. Set DMA address from ADDRESS.

SET-DMA

Sends the DMA address contained in EXT-ADDRESS to the disk controller, to use for the next operation.

SET-DENSE

Used when selecting a disk drive. Reads a sector id to determine the disk density, then sets the drive parameters accordingly.

SET-DRIVE

Initialize a newly selected disk drive. HOMEs the drive, SEEKs track 1, performs SET-DENSE, and sets OFFSET.

SET-ID

If the System Support 1 is available, set the EDITOR ID for the user's initials (from WHO) and the date. This routine becomes the normal value for BOOT, so it also performs SPECIFY to change the stepping rate for the disks.

SET-TIME

Set the clock on the System Support 1.

SHADOW n1 -- n2

n2 is the screen number of the shadow screen for screen n1.

SIDE -- adr

A variable containing the side of the disk to read or write.

SIGN n d -- d

Stores an ASCII "-" sign just before a converted numeric output string in the text output buffer when n is negative. n is discarded, but double number d is maintained. Must be used between <# and #>.

SIS -- f

Sense Interrupt Status of the disk controller. Returns flag; true if HOME or SEEK was successful.

SKEW -- adr

An array of variables containing the addresses of the skew tables for each drive. A zero address means no sector skewing. adr is a function of DRV.

SKIP n --

Sets SKIPPED to n.

SKIPPED -- adr

A variable containing the number of screens from the source to the destination in a multiple screen copy. Used by CONVEY.

SM adr --

Set Memory starting at adr; displays the byte contained at each address, accepts new values until a dot is entered. Entering CR leaves the value unchanged and continues.

SMUDGE

Used during word definition to toggle the "smudge bit" in a definition's name field. This prevents an uncompleted definition from being found during dictionary searches, until compiling is completed without an error. Without SMUDGE, all definitions would be recursive.

SP@ -- adr

Returns the contents of the stack pointer.

SP!
Initialize the stack pointer register from S0.

SPACE
Transmit an ASCII blank to the output device.

SPACES n --
Transmit n ASCII blanks to the output device.

SPECIFY
Send the disk controller a specify command, selecting the drive stepping rate. System boots with 8 milliseconds per step, and SPECIFY changes the rate to 4 milliseconds per step.

STAT -- n
A constant returning the I/O port number of the disk controller status port.

STATBUF -- adr
A buffer where status information from the disk controller is kept.

STATE -- adr U
A user variable containing the compilation state. A non-zero value indicates compilation.

SR! n --
Sets the 68000's status register to n. This is a privileged instruction.

SR@ -- n
Returns the contents of the 68000's status register.

SWAP n1 n2 -- n2 n1
Exchange the top two values on the stack.

SYS-SIZE -- adr
A variable containing the size of the system. Set by SAVE-SYSTEM

T&SCALC n --
Track and sector calculation for disk I/O. n is the total sector displacement from the start of the selected drive.

TRACK and SECTOR are set and a SEEK is performed to move the head to the correct track. For double sided disks, SIDE is set.

TASK

A no-operation word which can mark the boundary between applications. By forgetting TASK and re-compiling, an application can be discarded in its entirety.

TEXT c --

Reads a string from the input stream using the character c as a delimiter, then sets PAD to blanks and moves the string to PAD.

THRU u1 u2 --

LOAD blocks u1 through u2 inclusive.

THEN adr n -- (compiling) I,C

Used in a colon definition, in the form:

```
flag IF ... ELSE ... THEN
```

or

```
flag IF ... THEN
```

THEN is the endpoint of the conditional structure. During compilation, n is used for error checking and adr is used to resolve branching.

TIB -- adr U

A user variable containing the address of the text input buffer.

TIME

Prints the time of day.

TIMES n --

Causes the text input buffer up to TIMES to be re-interpreted n times. Used in the form:

```
xxx n TIMES
```

TO

Sets SKIPPED for a multiple screen copy. See CONVEY.

TOGGLE adr b --

Complement the contents of adr by the bit pattern b.

TRACK -- adr

A variable containing the disk track most recently accessed.

TRAVERSE adr1 n -- adr2

adr2 is the location of the first byte with its most significant bit set, after moving from adr1 by increment n. Usually n is +1 or -1. Typically used to move across a name in a dictionary entry.

TRIAD scr --

Display on the selected output device the three screens which include that numbered scr, beginning with a screen evenly divisible by three.

TRK/DRV — adr

An array of variables containing the number of tracks per drive for each disk drive. adr is a function of DRV.

TRUE -- true

A constant returning a true flag.

TYPE adr n --

Transmit n characters from adr to the selected output device.

TYPE. adr n --

TYPE n characters, replacing control characters with dots. Uses EMIT. and is used by DUMP.

U* u1 u2 — ud

Leave the unsigned double number product of two unsigned numbers.

U. u —

Type the value of an unsigned 16 bit integer.

U.R u n --

Type the value of the unsigned integer u, right justified in a field of n blanks.

U/D -- adr

A variable containing a direction flag for multiple screen copies. Used by CONVEY.

U/MOD ud ul -- u2 u3

Leave the unsigned remainder u2 and unsigned quotient u3 from the unsigned double dividend ud and unsigned divisor ul.

U< ul u2 -- f

Leave the boolean value of an unsigned less-than comparison. Leaves f=1 for ul < u2, otherwise leaves f=0. This function must be used when comparing memory addresses. ul and u2 are unsigned 16 bit integers.

UD. ud --

Type the value of the unsigned double number ud.

UD.R ud n --

Type the value of the unsigned double number ud, right justified in a field of n blanks.

UNTIL f -- (run-time) I,C
adr n -- (compile)

Occurs within a colon-definition in the form:

BEGIN ... UNTIL

At run-time, UNTIL controls the conditional branch back to the corresponding BEGIN. If f is false, execution returns to just after BEGIN, if f is true, execution continues ahead. At compile-time, UNTIL compiles ?BRANCH and an offset from HERE to addr. n is used for error testing.

UPC c1 -- c2

Converts character c1 to upper case.

UPDATE

Marks the most recently referenced block as altered. The block will subsequently be transferred automatically to disk should its memory buffer be required for storage of a different block or upon execution of FLUSH.

UPWORD c -- adr

Replaces WORD in -FIND. Converts input string to upper case before searching the dictionary or crecating a new word.

USER n --

A defining word used in the form:

n USER <name>

which creates a user variable <name>, whose parameter field contains n as a fixed offset relative to the user pointer. When <name> is later executed, it places the sum of its

offset and the user area base address on the stack.

VARIABLE

A defining word used in the form:

VARIABLE <name>

A dictionary entry for <name> is created and two bytes are ALLOTed in its parameter field to contain the contents of the variable. When <name> is executed, the address of its parameter field is placed on the stack.

VCREATE

A new version of CREATE which compiles the block number of the source code for a word in front of its dictionary entry for use by VIEW.

VIEW

Used in the form:

VIEW <name>

Lists the source screen for <name>, if <name> was created by VCREATE.

VOC-LINK — adr U

The address of a variable containing the address of the head of a linked list of vocabularies. All vocabularies are linked to allow FORGETting through multiple vocabularies.

VOCABULARY

A defining word used in the form:

VOCABULARY <name>

A dictionary entry for <name> is created which specifies a new ordered list of word definitions. Subsequent execution of <name> makes it the CONTEXT vocabulary. When <name> becomes the CURRENT vocabulary (see DEFINITIONS), new definitions will be created in that list.

WAIT-INT

Wait for the disk controller to complete execution of a command.

WARM

The warm start routine.

WARN pfa --

Displays the name of the word being defined and the 'not unique' message. Used by CREATE.

WHEN

Prints the time and date.

WHERE s n --

An execution vector, usually set to (WHERE). Used by ABORT" to display the source screen where an error occured during compilation.

WHILE f -- (execution) I,C
adr1 n1 -- adr1 n1 adr2 n2 (compilation)

Occurs in a colon-definition in the form:

BEGIN ... WHILE ... REPEAT

At run-time, WHILE selects conditional execution based on boolean flag f. If f is true (non-zero), WHILE continues execution of the true part through to REPEAT, which then branches back to BEGIN. If f is false (zero), execution skips to just after REPEAT, exiting the structure. At compile time, WHILE emplaces ?BRANCH and leaves addr2 of the reserved offset. The stack values will be resolved by REPEAT.

WHO

Prints the initials of the user.

WIDTH -- adr U

The address of a variable containing the maximum number of characters to be saved in the name of a dictionary entry. (1..31)

WITHIN n1 n2 n3 -- f

Returns a flag; true if n1 is between n2 and n3.

WORD c -- here

Read the next text characters from the input stream being interpreted, until a delimiter c is found, storing the packed character string beginning at the dictionary buffer HERE. WORD leaves the character count in the first byte, the characters, and ends with two or more blanks. Leading occurrences of c are ignored. If BLK is zero, text is taken from the terminal input buffer, otherwise from the disk block stored in BLK. See BLK, IN.

WORDS

List the names of the words in the CONTEXT dictionary search order.

XOR n1 n2 -- xor

Leave the bitwise logical exclusive or of two 16-bit values.

[I

Used in a colon-definition in the form:

```
: xxx [ words ] more ;
```

Suspend compilation. The words after [are executed, not compiled. This allows calculation or compilation exceptions before resuming compilation with]. See LITERAL,].

[´] I,C

Used only inside a colon definition in the form:

```
[´] <name>
```

Compiles the parameter field address of <name> as a literal.

[COMPILE] I,C

Used in a colon-definition in the form:

```
: xxx [COMPILE] <name> ;
```

[COMPILE] will force the compilation of an immediate word, which would otherwise execute during compilation.

**** I

Comment to end of line. May only be used on disk.

]

Resume compilation, to the completion of a colon-definition. See [.

PURPOSE

The primary purpose of an assembler in a Forth environment is to allow writing Forth words in machine code. It is not intended to create stand alone assembler applications. Ordinarily, stand alone applications are produced with a meta compiler or a cross assembler. mapForth was produced with a meta compiler, but the meta compiler is not yet a product.

In writing applications, the assembler is rarely used until the algorithms have been tested and debugged in high level code. At the initial stages of a design, the programmer's time is far more valuable than the machine's. When an application is running, it might prove to be too slow. If so, you now decide where the most time is being spent, and rewrite just that routine in code. Repeat this process until the application is fast enough. Avoid writing in code unless necessary: it limits portability. Almost all the mapForth utilities were running on an 8080 Forth system before they ran on the 68000, a very few changes were needed to move them from one CPU to another.

USE

As an example of how the assembler is used, take the definition of the word FILL, which fills an area of memory with a given byte. It is used in the form.

```
address length byte FILL
```

Notice that FILL takes three parameters from the stack and does not return any. The definition of FILL is:

```
CODE FILL  (S adr len byte -- )
    SP )+ D0 MOVE ( pop 'byte' into D0 )
    SP )+ D1 MOVE ( pop 'len' into D1 )
    SP )+ D7 MOVE ( pop 'adr' into D7 whose high half is 0 )
    D7 A0 LMOVE   ( move all of D7 into A0 )
    1 D1 SUBQ     ( decrement D1: DBRA goes to -1, not to 0 )
    D1 D0  D0 A0 )+ BYTE MOVE   LOOP   (loop until D1 is -1,
         each time moving the byte in D0 to the address in A0,
         and incrementing A0 )
    NEXT   ( compile a jump to 'next' )
    C;     ( end definition )
```

IMPLEMENTATION

The assembler is based on the word ´,´ (comma), which appends a number to the dictionary (at HERE). The work CODE CREATEs a word whose code field points to its own parameter field, and leaves the system in execution state: the assembler does not use the compiler. Assemble opcode mnemonics such as MOVE use comma to add numbers in line to the parameter field of the word being defined. If the new word is to return to Forth after executing, its definition will end with NEXT. NEXT is a macro which assembles a jump to Forth´s address interpreter. Its definition is:

```
: NEXT  (NEXT) #) JMP;
```

where (NEXT) is the address of the interpreter, #) indicates the ´absolute word address´ mode, and JMP uses comma to append the proper opcode and address to the word being defined.

For more information, see the assembler source code and shadow screens.

mapFORTH

A *GODBOUT* COMPANY
3506 Breakwater Court, Hayward, CA 94545